CS 3430: Python & Perl
Coding Exam 2

Vladimir Kulyukin
Department of Computer Science
Utah State University

Instructions

- There is one coding problem on this exam. You have to solve it in both Py and PL. I have broken the problem into four smaller sub-problems.

- You may use any online and paper references, including the class lecture pdfs and code samples at http://vkedco.blogspot.com/2012/01/python-and-perl-programming-home.html). If you copy and paste a chunk of code from a site, I would appreciate a quick reference in your comments.

- There are two attached files, small_access_log.txt and access_log.txt, that you will use as input for this exam. I have edited them on DOS. If you are on Linux/Free BSD, etc use dos2unix to convert the format.

- Write your first and last name and your A# at the beginning of each file.

- Your solutions are due by 10:30am. I will use the following formula for deducting points for late submissions: $f(x) = 2^x$, where x is the number of minus past 10:30am.

- In addition to submitting your solutions via Canvas, please email them to vladimir.kulyukin@usu.edu as an attached with the subject header: CS3430 Coding Exam 2 and your name and A# in the body of the message. Some of you had trouble submitting your coding exam solutions via Canvas. So, if I do not see your Canvas submission, I will grade files in your email attachment.

- Happy Hacking!

Problem 1 (15 points)

There are two attached Apache server web logs small_access_log.txt and access_log.txt attached. These are the input files you will be working on. These are real log files I downloaded from http://www.monitorware.com/en/logsamples/apache.php. You should be closely familiar with the formal of web logs from the previous assignment so, for the sake of time, I will skip it here.

Subproblem 1.1 (2 points for Py & 2 points for PL)

Write Py and PL scripts fltr_tabbed_recs.py and fltr_tabbed_recs.pl that take access_log.txt or small_access_log.txt and output and splits each line into tab-separated line records. Each record consists of date, time, method, ip, transferred bytes, and result status. Below is the command line output your scripts should generate on small_access_log.txt. You must use regular expressions to split each web log entry into groups and both scripts must take their input from STDIN and output into STDOUT.

```
>type small_access_log.txt | python fltr_tabbed_recs.py
```

```
07/Mar/2004	16:05:49	GET	64.242.88.10	12846	401
07/Mar/2004	16:06:51	GET	64.242.88.10	4523	200
07/Mar/2004	16:10:02	GET	64.242.88.10	6291	200
07/Mar/2004	16:11:58	GET	64.242.88.10	7352	200
07/Mar/2004	16:20:55	GET	64.242.88.10	5253	200
07/Mar/2004	16:23:12	GET	64.242.88.10	11382	200
07/Mar/2004	16:24:16	GET	64.242.88.10	4924	200
07/Mar/2004	16:29:16	GET	64.242.88.10	12851	401

>perl fltr_tabbed_recs.pl small_access_log.txt
07/Mar/2004	16:05:49	GET	64.242.88.10	12846	401
07/Mar/2004	16:06:51	GET	64.242.88.10	4523	200
07/Mar/2004	16:10:02	GET	64.242.88.10	6291	200
07/Mar/2004	16:11:58	GET	64.242.88.10	7352	200
07/Mar/2004	16:20:55	GET	64.242.88.10	5253	200
07/Mar/2004	16:23:12	GET	64.242.88.10	11382	200
07/Mar/2004	16:24:16	GET	64.242.88.10	4924	200
07/Mar/2004	16:29:16	GET	64.242.88.10	12851	401
```

Subproblem 1.2 (1 point for Py & 1 point for PL)

Write Py and PL scripts fltr_binary_toops.py and fltr_binary_toops.pl that take the tab-separated records from fltr_tabbed_recs defined in sub-problem 1.1 and two command line integer arguments specifying the column numbers in of each tab-separated entry. The scripts take their input from STDIN and output into STDOUT the binary tab-separated toops where the first element is the value of the first column in each tab-separated entry and the second element is the value of the second column in each tab-separated entry. Those of you familiar with SQL will immediately notice that we are computing table projections. Below is the output for both Py and PL computing the project ion of ips and transferred bytes.

```
>type small_access_log.txt | python fltr_tabbed_recs.py | python fltr_binary_toops.py 3 4
64.242.88.10	12846
64.242.88.10	4523
64.242.88.10	6291
64.242.88.10	7352
64.242.88.10	5253
64.242.88.10	11382
64.242.88.10	4924
64.242.88.10	12851

>fltr_tabbed_recs.pl small_access_log.txt | fltr_binary_toops.pl 3 4
64.242.88.10	12846
64.242.88.10	4523
64.242.88.10	6291
64.242.88.10	7352
```

64.242.88.10 5253
64.242.88.10 11382
64.242.88.10 4924
64.242.88.10 12851

Subproblem 1.3 (2.5 points for Py & 2.5 points for PL)

Write Py and PL scripts count_binary_toops.py and count_binary_toops.pl that take binary tabbed-separated toops produced by fltr_binary_toops. They treat each binary tab-separated toop as a key-value pair where the first element is a key and the second element in an integer. They use hash tables (Py dictionary or PL hash) to output the total key sum for each key. Again, the input comes from STDIN and the output from STDOUT. Below is the output of both scripts on small_access_log.txt. Each script essentially sums up how many bytes were transferred by 64.242.88.10.

>type small_access_log.txt | python fltr_tabbed_recs.py | python fltr_binary_toops.py 3 4 | python count_binary_toops.py
64.242.88.10 65422

>fltr_tabbed_recs.pl small_access_log.txt | fltr_binary_toops.pl 3 4 | count_binary_toops.pl
64.242.88.10 65422

Subproblem 1.4 (2 points for Py & 2 points for PL)

Write Py and PL scripts desc_sort_binary_toops.py and desc_sort_binary_toops.pl that take binary tab-separated toops produced by count_binary_toops and sort them in descending order by the value of the second element in each binary toop. The input for each script is STDIN and the output is STDOUT. You do not have to implement top_n. It is used just to illustrate the top 5 entries in the sorted list that should be output into STDOUT. Your PL solution must use the stable quicksort.

C:\Users\Vladimir\programming\py\coding_exam_02>type access_log.txt | python fltr_tabbed_recs.py | python fltr_binary_toops.py 3 4 | python count_binary_toops.py | python top_n.py 5
212.21.228.26 2869
dialup-5-81.tulane.edu 27978
lj1028.inktomisearch.com 209
ladybug.cns.vt.edu 24099
10.0.0.153 1200145

>fltr_tabbed_recs.pl access_log.txt | fltr_binary_toops.pl 3 4 | count_binary_toops.pl | desc_sort_binary_toops.pl | top_n.pl 5
64.242.88.10 5745035
10.0.0.153 1200145
cr020r01-3.sac.overture.com 627328
h24-71-236-129.ca.shawcable.net 250905
h24-70-69-74.ca.shawcable.net 150461

What to Submit

Create a zip archive coding_exam_2.zip and submit it via Canvas and email it to me as an attachment. Your zip archive should contain fltr_tabbed_recs.py and fltr_tabbed_recs.pl, fltr_binary_toops.py and fltr_binary_toops.pl, count_binary_toops.py and count_binary_toops.pl, desc_sort_binary_toops.py and desc_sort_binary_toops.pl.

Happy Hacking